University of California
Santa Barbara

Things and Strings and More: Improving Place Name Disambiguation from Short Texts by Combining Entity Co-Occurrence, Topic Modeling, and Word Embedding

A thesis submitted in partial satisfaction
of the requirements for the degree

Master of Arts
in
Geography

by

Yiting Ju

Committee in charge:

Professor Krzysztof W. Janowicz, Chair
Professor Werner Kuhn
Professor Konstadinos Goulias

December 2017

The Thesis of Yiting Ju is approved.

Professor Werner Kuhn

Professor Konstadinos Goulias

Professor Krzysztof W. Janowicz, Committee Chair

September 2017

Things and Strings and More:

Improving Place Name Disambiguation from Short Texts by Combining Entity Co-Occurrence, Topic Modeling, and Word Embedding

by

Yiting Ju

Acknowledgements

Gratitude to my family, my friends, my teachers, and every member of the STKO Lab.

Abstract

Things and Strings and More:
Improving Place Name Disambiguation from Short Texts by Combining Entity Co-Occurrence, Topic Modeling, and Word Embedding

by

Yiting Ju

Place name disambiguation, i.e., toponym disambiguation or toponym resolution, is the task of correctly identifying a place from a set of places sharing a common name. It contributes to a variety of tasks such as knowledge extraction, query answering, geographic information retrieval, and automatic tagging. Disambiguation quality relies on the ability to correctly identify and interpret contextual clues, complicating the task for short texts. Here I propose a novel approach to the disambiguation of place names from short texts that integrates three models: entity co-occurrence, topic modeling, and word embedding. The first model uses Linked Data to identify related entities to improve disambiguation quality. The second model uses topic modeling to differentiate places based on the terms used to describe them. The third model uses word embeddings to uncover the semantic relatedness between places and contexts. I evaluate this approach using a corpus of short texts collected through web scraping, determine the suitable weights for the models, and demonstrate that the combined model, i.e., Things and Strings Model, outperforms benchmark systems such as DBpedia Spotlight, TextRazor, and Open Calais by up to 85% in F-score and 46% in Precision at 1. A web service is built to demonstrate the proposed method and it can be a building block for those applications that need place name recognition and disambiguation.

Contents

Abstract **v**

1 Introduction **1**

2 Related Work **6**

3 Methodology **9**
3.1 Entity-based Co-Occurrence Model 10
3.2 Topic-based Model 12
3.3 Word Embedding Model 16
3.4 Things & Strings Model (TSM) 20

4 Evaluation **23**
4.1 Preparing the Test Corpus 23
4.2 Metrics 25
4.3 Benchmark systems 26
4.4 Results 27

5 Discussion and Conclusions **35**

Bibliography **39**

Chapter 1

Introduction

Geographic knowledge extraction and management, geographic information retrieval, question answering, and exploratory search hold great promise for various application areas [1, 2, 3]. From intelligence and media analysis to socio-environmental studies and disaster response, there is demonstrated need to be able to build computational systems that can synthesize and understand human expressions of information about places and events occurring around the world [4]. In today's Big Data era, an enormous amount of information is generated every second, most of which is composed of unstructured plain texts. Being able to efficiently and correctly identify geographic references in the abundance of such textual information now available on the Web, in social media, and in other communication media is the first step to building tools for geographic analysis and discovery on these data. Place name, i.e., toponym, disambiguation is key to the comprehension of many texts as place names provide an important context required for the successful interpretation of text [5].

Similar to other named entities, including persons, organizations, and events, place names can be ambiguous. A single place name can be shared among multiple places. To give a concrete example, *Washington* is a place name for more than 43 populated places in the United States alone.[1] Although most of these Washingtons can be accurately

[1]https://en.wikipedia.org/wiki/Washington

located by adding the proper state name or county name, they are all simply referred to as *Washington* in daily conversations, (social) media, photo annotations, and so forth. Figure 1.1 depicts the distribution of the most common place names for U.S. cities, towns, villages, boroughs, and census-designated places. As shown on the map, these places are distributed across the U.S., indicating that the ambiguity of place names is a widespread phenomenon. It is worth noting that places which share a common name can be of the same or a different type, e.g., the *state* of Washington and the *city* of Washington, Pennsylvania. The situation is even more difficult on a global scale where place names may appear more than 100 times. For example, it takes merely a 45min car ride to get from Berlin to East London, both located in South Africa. Thus, it is important to devise effective computational approaches to address the disambiguation problem.

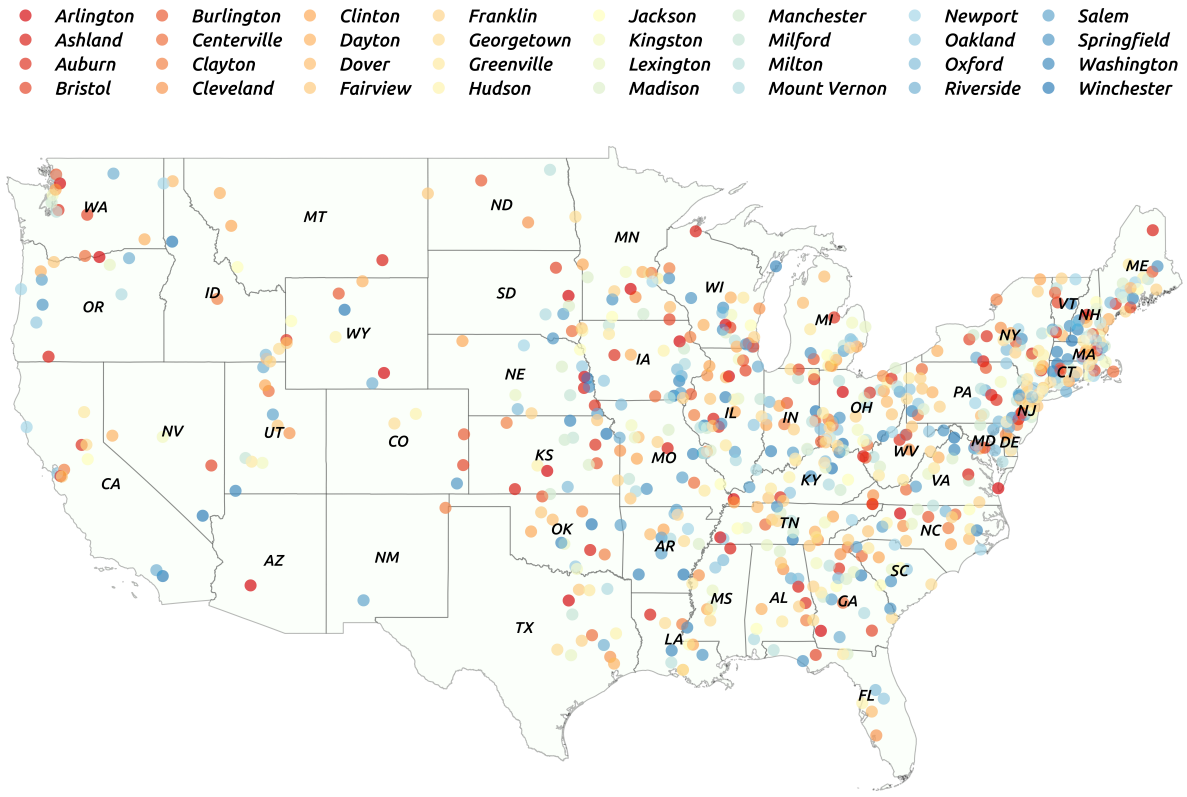

Figure 1.1: Distribution of common place names in the US according to Wikipedia.

Given the wide availability of digital gazetteers, i.e., place name dictionaries, such as GeoNames, the Getty Thesaurus of Geographic Names, the Alexandria Digital Library Gazetteer, and Google Places, I assume that the places to be disambiguated are known, i.e. that there is a candidate list of places for any given place name list. After all, unknown places cannot be disambiguated. Thus, I define the task of place name disambiguation as follows: *given a short text which contains a place name and given a list of candidate places that share this name, determine to which specific place the text refers.*

Humans are very good at detecting and interpreting contextual clues in texts to disambiguate place names. Thus, as extension of named entity recognition, place name disambiguation has been tackled using computational approaches that aim at utilizing these contextual clues as well [6, 7]. This context typically stems from the terms surrounding the place name under consideration. Typically, short texts from social media, news headlines (and abstracts), captions, and so forth, offer less contextual clues and thus negatively impact disambiguation quality. Consequently, new approaches have to be develop that can extract and interpret other contextual clues.

One such approach is to focus on the detection of surrounding *entities* and use these as contextual clues. Besides the place itself, these entities may include other places, actors, objects, organizations, and events. Examples of such associated entities are landmarks, sports teams, well known figures such as politicians or celebrities, and nearby places that share a common administrative unit [8]. Intuitively, when a text mentions *Washington* along with *Redskins*, an American football team based in Washington, D.C., it is very likely that the *Washington* in the text refers to Washington, D.C., rather than the other places called Washington. It has been shown that such a co-occurrence model increases disambiguation quality [9, 10].

In addition to entities, implicit *thematic* information buried in the text can also provide contextual evidence to disambiguate place names. Similar to entities, some par-

ticular topics are more likely to be mentioned along with a place, which is characterized by those topics. Topic modeling makes it possible to discover topics from the text and match texts with similar topics. Thus, given topics learned from a corpus of texts about candidate places and the topics discovered from the short text under consideration, computing a similarity score between the topics representative for the text and for each of the candidate places can provide additional contextual clues [11]. For example, when people are talking about *Washington, DC*, political topics featuring terms such as *conservative*, *policy*, and *liberal* are more likely to be mentioned than when talking about the (small) city of *Washington, Pennsylvania*.

Besides topics, other implicit information embedded in the text can also carry useful clues to facilitate place name disambiguation. Such implicit information can be further exploited through word embedding models, a language model representing words in vectors of real numbers, which capture both syntactic and semantic word relationships [12]. Through a trained word embedding model, vectors can be learned from those paragraphs describing the candidate places. Meanwhile, another vector representation can be generated to represent the short text under consideration. The similarity scores between the vector representation of the short text and the vector representation of each of the candidates places can infer which candidate place is more likely to be the actual place discussed in the short text.

The core distinction between these perspectives is that mentioned entities are explicit information, while thematic information is usually implicit. Both explicit and implicit information in the contexts is used as clues by humans to disambiguate place names. In this thesis, I propose a novel approach which integrates *things and strings*, i.e., entity co-occurrence, topic modeling and word embedding, thereby combining explicit and implicit contextual clues. **The contributions of this work are as follows:**

- I apply word embedding to place name disambiguation, an approach that has not been taken before.

- I integrate a topic-based model and a word embedding model with a reworked version of the entity-based co-occurrence model and learn the appropriate weights for this integrated model.

- I compare the integrated model to three well known systems (TextRazor, DBpedia Spotlight, and Open Calais) as benchmark systems and demonstrate that my model outperforms all of them.

- I share the source code on GitHub[2] and create a web service to make this work reproducible and to provide a new benchmark for future studies

The remainder of this thesis is organized as follows. Chapter 2 reviews related work on place name disambiguation. Chapter 3 describes the methodologies I apply to disambiguate place names. The evaluation and results are illustrated in Chapter 4. Finally, Chapter 5 concludes the research and discusses the potential applications and limitations of the proposed method.

[2]https://github.com/DajeRoma/Things_and_Strings

Chapter 2

Related Work

In this section, I review existing research that involves place name disambiguation in natural language.

As an extension of named entity disambiguation, place name disambiguation can be conducted using the general approaches for named entity disambiguation. Generally, named entity disambiguation is constructed as a ranking problem in which entities are assigned to its most similar Wikipedia page. Wikipedia is also served as a valuable source for grounding truth descriptions of named entities in a number of studies. Bunescu and Pasca [6] trained a vector space model to host the contextual and categorical terms derived from Wikipedia and employed TF-IDF to determine the importance of these terms. The system defined a similarity threshold below which no place entity would be assigned. Cucerzan [13] also applied the vector space model with Wikipedia. The author considered the other entity references in the same document as the context for each entity that was being disambiguated, in addition to the surrounding words. The vector for the text contained the categories of all possible referents for all entity references found in the text and the occurrences of each references. Each referent had a binary feature vector with all the categories and entity references found in the corresponding Wikipedia page. The similarity was then measured while the feature values were not normalized, which privileged important entities with long descriptions, more mentions, and more categories.

Milne and Witten [14] described a method for augmenting unstructured text with links to Wikipedia articles. For ambiguous links, the authors proposed a machine learning approach and trained several models based on Wikipedia data. Two named entity disambiguation modules were introduced by Mihalcea and Csomai [15]. One measured the overlaps between context and candidate descriptions and the other trained a supervised learning model based on manually assigned links in the Wikipedia articles. Zheng et al. [16] evaluated ranking methods for linking entities, which included point-wise, pair-wise, and list-wise ranking algorithms. Eighteen features were employed, covering features from surface names, features from contexts, and mentioned countries or cities in the text.

For previous studies focusing on disambiguating place names, Leidner [17] had a survey of the related methods, finding most methods relied on heuristic rules, such as always assigning the most important candidate locations (e.g., largest population) and generating a minimal bounding polygon to contain as many candidate locations as possible. Jones and Purves [5] discussed using co-occurred places in the text to resolve place ambiguity. Machado et al. [18] proposed an ontological gazetteer which recorded the semantic relations between places to help disambiguate place names based on related places and alternative place names. In a similar approach, Spitz et al. [8] constructed a network of place relatedness based on the English Wikipedia articles.

Machine learning approaches got increasing attention to tackle information retrieval problems, same for place name disambiguation. Lieberman and Samet [19] trained a binary random forest classifier for each candidate locations with a large set of features called adaptive context features. It should be noted that they took geospatial proximity and geographic hierarchy between place names into consideration. Speriosu et al. [20] trained a couple of classifiers based on contextual information using geotagged Wikipedia articles. Zhang and Gelernter [21] proposed a supervised machine learning approach to

rank candidate places for ambiguous toponyms in Twitter messages that relied on the metadata of tweets and context to a limited extent. De Bruijn et al. [22] took temporal contexts into account for geotagging tweets.

In the previous work, Hu [9] leveraged the structured Linked Data in DBpedia for place name disambiguation and demonstrated that a combination of Wikipedia and DBpedia data leads to generally better performance. This work is an extension of the previous work of my colleagues and me [23], in which we introduced topic model and integrated it with entity co-occurrence model to tackle place name disambiguation.

Chapter 3

Methodology

The work at hand differs from these previous studies. I apply topic modeling as well as word embeddings to capture implicit semantic clues embedded in short texts, and integrate them with an entity-based model which uncovers the co-occurrence relations among entities. Thereby, I combine a *strings*-based perspective with a *things*-based perspective.

Since the work focuses on disambiguating place names, I assume that the *surface forms* of place names have been extracted prior to disambiguation, so the primary task is to identify the place to which the surface form refers. To accomplish this, a list of candidate entities, i.e., places, is selected. In prior work, knowledge bases, such as Wikipedia, DBpedia, and WordNet have been used to obtain candidate entities [13, 24, 25], and here I use DBpedia as the source of candidate locations. Once a set of candidate locations has been identified, the likelihood that the surface form refers to each candidate location can be measured. The disambiguation result is returned if the computed likelihood score exceeds a given threshold.

In the following sections, I describe each of the three models I employ to disambiguate place names, followed by a depiction of how I integrate these models.

3.1 Entity-based Co-Occurrence Model

In this section I describe the entity-based co-occurrence method. Wikipedia and DBpedia are used as the sources to train my model. I define the entities from Wikipedia as those words or phrases on a Wikipedia page of the candidate location which have links to another page about these entities. The entities from DBpedia are either the subjects or the objects of those RDF triples which contain the candidate entities. Not all RDF triples are selected, but those that fall under the DBpedia namespace, i.e., with prefix *dbp*[1] and *dbo*.[2] While *dbo* provides a cleaner and better structured mapping-based dataset, it does not provide a complete coverage of the original properties and types from the Wikipedia infoboxes. In order to avoid data bias, I use both *dbo* and *dbp*. Literals were excluded. I treat the subject and the object of a triple as a whole, i.e., as an individual entity, instead of further tokenizing them into terms. The harvested entities differ greatly. They include related places (of same or different types), time zone information, known figures that were born or died at the given place, events that took place there, companies, organizations,[3] sports teams, as well as representative landmarks such as buildings or other physical objects.

Table 3.1 shows several sample entities for Washington, Louisiana, derived from Wikipedia and DBpedia. It should be noted that there is considerable overlap between place data extracted from Wikipedia and DBpedia. Moreover, some properties such as *population density* in Wikipedia can occur for most or even all candidate locations. Such entities which appear frequently but help less to uniquely identify a place will not play a crucial rule in disambiguating the place names.

The entities are assigned weights according to their relative connectivity to the places

[1] http://dbpedia.org/resource/
[2] http://dbpedia.org/ontology/
[3] For example via dbr:FreedomWorks dbp:headquarters dbr:Washington,_D.C. .

Washington, Louisiana
Wikipedia — St.Landry Parish; Opelousas; Eunice; population density; medianhousehold income; American Civil War; Connecticut; cattle; cow; corn...
DBpedia — United States; Central Time Zone; St. Landry Parish, Louisiana; John M. Parker; KNEX-FM; Louisiana Highway 10...

Table 3.1: Sample entities for Washington, LA, from Wikipedia and DBpedia

by means of *term frequency-inverse document frequency (TF-IDF)*. The term frequency of an entity is the number of times the entity appears in Wikipedia and DBpedia, so in this case, it could be 0, 1, and 2. I only count each entity's appearance in a document once, so the term frequency will not be inflated by those entities which are related to many candidate entities while contribute less to uniquely identifying the place. The formula of applying TF-IDF to assign weights to entities is defined in Eq. 3.1, 3.2, and 3.3.

$$tf(e) = \begin{cases} 0 & e \text{ is not in Wikipedia and DBpedia} \\ 1 & e \text{ is either in Wikipedia or DBpedia} \\ 2 & e \text{ is in both Wikipedia and DBpedia} \end{cases} \tag{3.1}$$

$$idf(e) = 1 + log(\frac{|E|+1}{n_e}) \tag{3.2}$$

$$Weight(e) = TF\text{-}IDF(e) = tf(e) \times idf(e) \tag{3.3}$$

where $tf(e)$ defines the term frequency of an entity e, and $idf(e)$ defines the inverse document frequency of e. $|E|$ is the number of all potential candidate locations for a surface form, and n_e represents the number of candidates which contain the entity e. Using TF-IDF, entities appearing in multiple candidates are given lower weights, while

entities which are able to uniquely identify a place have higher weights. For example, the fact that a place is within the United States becomes irrelevant as it holds for all of them.

I then measure the likelihood that a surface form in a test sentence refers to a candidate location through an entity matching score. To compute the entity matching score, I first find all the entities of the candidate which also appear in the short text. The weights of matching entities are summed to produce an entity matching score of the candidate location to the surface form in the test sentence. The score is calculated as given in Eq. 3.4.

$$S_{EC}(p \rightarrow c_i) = \sum_{j=1}^{m} (Weight(e_j) \times I_j) \tag{3.4}$$

where m corresponds to the number of entities e for the candidate c_i. I_j is either 1 or 0, referring to whether a matching entity is found in the test for the entity e_j. The candidate location with higher entity matching score is regarded to more likely be the actual place p to which the surface form refers. The matching score is the final output of the entity co-occurrence model.

3.2 Topic-based Model

In this section I introduce the topic-based model. It makes use of the fact that text is *geo-indicative* [11] even without having any direct geographic references. Hence, even everyday language should be able to provide additional evidence for place name disambiguation. For example, terms such as *humid*, *hot*, *festival*, *poverty*, and even *American Civil War* are more likely to be uttered when referring to Washington, Louisiana than Washington, Maine. The latter rarely experiences hot and humid weather, does not host

a popular festival, has substantially less poverty problems compared to its namesake, and did not play a notable role in the civil war.

Here I use Latent Dirichlet allocation (LDA) for topic modeling. LDA is a popular unsupervised machine learning algorithm used to discover topics in a large document collection [26]. Each document is modeled as a probability vector over a set of topics, providing a dimensionally-reduced representation of the documents in the corpus.

I use the geo-referenced text from the English Wikipedia as the source material for discovering these thematic patterns. I start with the idea that a collection of texts that describe various features in a local region–such as museums, parks, mountains, architectural landmarks, etc.–give us a foundation for differentiating places referenced in other texts based on thematic, non-geographically specific, terms. For this I need a systematic way to associate the training documents in Wikipedia with well-defined regions. Because administrative regions vary widely in area, they do not provide a good mechanism for aggregation. Instead, my solution is to aggregate the geo-referenced texts in Wikipedia based on an equal area grid over the Earth. This solution means that articles with point-based geo-references are binned together if they spatially intersect with a grid cell, while text related to areal features (such as national parks) can be associated with multiple grid cells.

There are several options for creating a discrete global grid based on an polyhedral simplification of the Earth [27]. In this work I utilize the Fuller icosahedral Dymaxion projection to create a hierarchical triangular mesh [28]. The triangular mesh can be made successively more fine-grained by dividing each triangle into four internal triangles. Therefore, the size of the grid cells can be relative to the other features on the map at all zoom levels [3]. For place name disambiguation I need grid cells that are fine-grained enough so that two possible places with the same name do not fall within one grid cell. The Fuller projection at hierarchical level 7 (shown in Figure 3.1) provides a mesh over

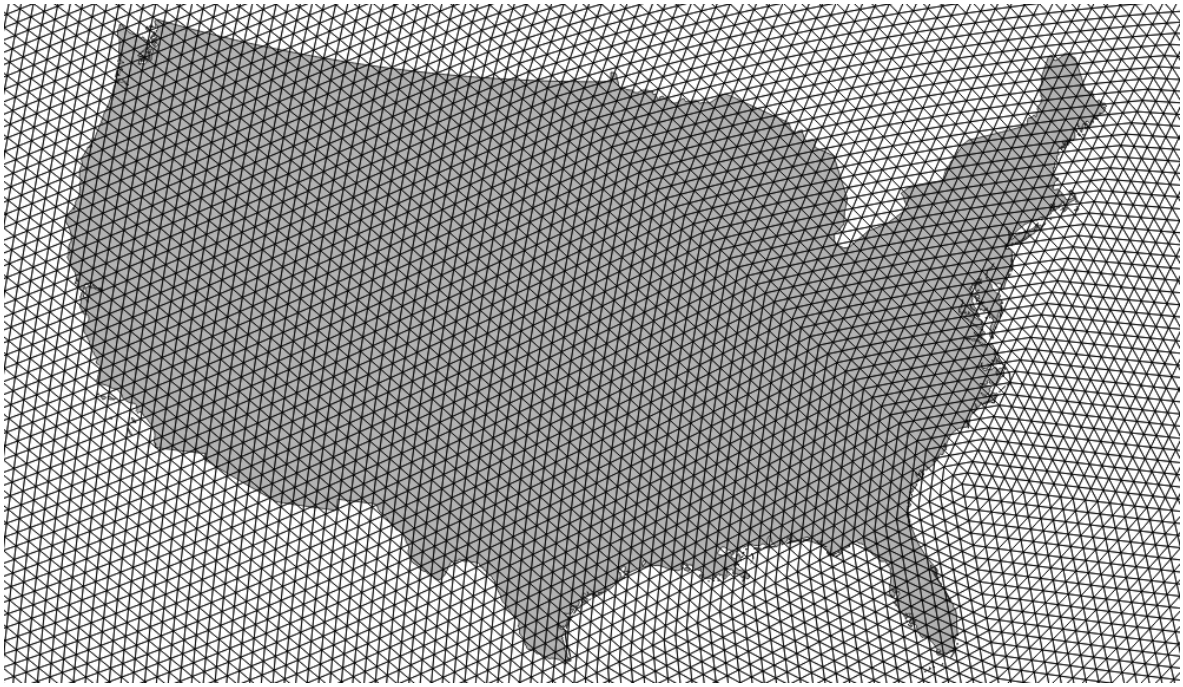

Figure 3.1: Level 7 triangular mesh discrete global grid built using Fuller icosahedral Dymaxion projection, shown in U.S. Contiguous Albers projection.

the Earth with 327,680 cells with inter-cell distance of 31.81 km and cell area of 1,556.6 km^2, sufficient to handle most place name disambiguation tasks for meso-scale features like cities.

Once I identified all articles that have geo-references that spatially intersect with a grid cell I can combine all the text to create a *grid document*. For the English Wikipedia the geo-referenced articles intersect with 63,473 grid cells at Fuller level 7. The resulting 63,473 grid documents serve as the training data input for LDA topic modeling. I utilized the MALLET implementation of LDA with hyperparameter optimization, which allows for topics to vary in importance in the generated corpus [29, 30].

For this work, I trained the LDA topic model with 512 topics. Figure 3.2 shows four sample topics, two of which are indicative of text from the grid cell containing the town of Washington, Illinois, and the other two of high probability for Washington, Louisiana. These topics demonstrate that not only do related place names help to disambiguate,

Figure 3.2: Sample topics that help differentiate between the towns of Washington, Illinois and Washington, Louisiana.

but also thematic words like 'hurricane' and geographic feature type terms like 'parish' help as well.

The MALLET toolkit generates an inferencer file for testing new documents against a trained LDA model. For a new document or snippet of text, I use the trained topic model to infer the most likely candidate location based on the inferred mixture of topics. Given a set of candidate locations (i.e., point coordinates) I find the topic mixtures for the grid cells that spatially intersect the locations and calculate the Jensen-Shannon divergence (Eq.3.6) between probability vector representations of the topic mixtures for each candidate and the topic mixture for the new document. The JS divergence is a symmetric measure calculated from the average of the relative entropies (Kullback Leibler divergence, shown in Eq. 3.5) between two probability vectors (P and Q) and their average, $M = \frac{1}{2}(P + Q)$. The JS divergence is a standard measure of similarity between two probability vectors, and is commonly used for calculating similarity based on topic model results [31]. A lower JS divergence result indicates greater thematic similarity between the new text and the candidate location.

$$KL(P \parallel Q) = \sum_i P(i) \log_2 \frac{P(i)}{Q(i)} \tag{3.5}$$

$$JS(P \parallel Q) = \frac{1}{2}KL(P \parallel M) + \frac{1}{2}KL(Q \parallel M) \tag{3.6}$$

Therefore, the matching score between a candidate location to the surface form can be formalized as Eq. 3.7.

$$S_{TM}(p \rightarrow c_i) = -JS(PV(c_i) \parallel PV'(t)) \tag{3.7}$$

Where $PV(i)$ refers to the probability vector associated to the grid cell where the candidate c_i is located and $PV'(t)$ is the probability vector inferred from the test sentence where the surface form p is. This matching score is the final output of the topic model.

3.3 Word Embedding Model

In this section I introduce how I approach place name disambiguation using word embedding models.

Word embeddings are a set of Natural Language Processing (NLP) feature learning techniques to represent words and phrases with vectors of real numbers. In linguistics, it belongs to the branch of distributional semantics, which studies methods of quantifying semantic similarity between linguistic items based on the assumption that linguistic items with similar distributions have similar semantics [32]. A better word embedding allows the vectors to carry more precise syntactic and semantic word and phrase relationships. The idea of representing words and phrases as continuous vectors was early discussed in [33, 34]. An influential work by Bengio, et al. [35] tackled NLP tasks with distributed representation of words which were learned through a neural probabilistic language model. Word embedding models showed significant improvement and simplification in many NLP applications such as sentiment analysis, image annotation, paraphrase detection,

automatic speech recognition and machine translation [36, 37, 38, 39, 40, 41].

To generate high-quality word embeddings, there are basically two primary model families, namely *count-based methods* and *prediction-based methods*. The count-based methods, represented by well-known Latent Semantic Analysis (LSA), compute the statistics of co-occurrence of a word with other words and map these statistics to a small dense vector for each word using dimension reduction techniques such as Singular Value Decomposition (SVD) [42]. The prediction-based methods, on the other hand, aim to predict a word based on its surrounding words or predict surrounding words based on the central word.

Despite the good performance of the word embedding models, they were too computationally expensive to be adopted widely until the recent introduction of *word2vec* [43, 12]. Word2vec is a word embedding framework composed of two major models, namely Continuous Bag-of-words Model (CBOW) and Continuous Skip-gram Model. Both models are new members to the prediction-based methods. While CBOW predicts the current word given the context, Skip-gram Model tries to predict the surrounding words given the current word. Unlike the previous models involving dense matrix multiplications and multiple layers of complex neural networks, both CBOW and Skip-gram model are simpler with only two layers of neural networks but they are significantly more efficient in learning high-quality word representations than former models.

The word2vec model outputs a set of vectors, each of which corresponds to a word or a phrase. Interestingly, linear regularities are found in the word vectors learned through word2vec. For example, the result of the vector calculation of *vec("Paris")* - *vec("France")* + *vec("Italy")* is closer to *vec("Rome")* than to other word vectors.

In addition to word2vec, *GloVe* is another word embedding model which can also generate high-quality vector representations [44]. It is an optimized combination of both count-based and prediction-based methods. Linear regularities can also be found in the

learned vectors from GloVe.

In order to better disambiguate place names, I utilize word embedding models (i.e., word2vec and GloVe) to capture the semantic relationship and relatedness embedded in the context. Specifically, I generate a vector representation for a test sentence which contains a surface form. For each candidate location of the surface form, I obtain vector representations of its English Wikipedia textual content. Calculating the similarities between these two vectors gives a clue of which candidate's description is semantically closer to the test sentence. Gensim, a Python framework for a number of semantic modelings, is employed for manipulating word embedding models [45].

Provided that in this case I am dealing with natural language items, i.e., normal words and phrases, and there are abundant word2vec and GloVe models which have been well trained on gigantic corpus of natural language, I directly employ those trained models for this study. The trained models are varied in the corpus, in the dimension of the trained vectors, and in the training methods (CBOW versus skip-gram versus GloVe). No research has shown which pre-trained model outperforms the others, though empirically high-dimension models contains richer semantics, and the corpus can also play a crucial role on the performance, especially for those tasks in a certain domain. To pick the word embedding model that is most suitable to this task, I compared five widely used models, namely a word2vec model trained on Google News in English of 3 million unique words or phrases with 300 dimensions by the Google Brain group developing word2vec, and four GloVe models trained on English Wikipedia and English Gigaword Fifth Edition with 50, 100, 200, 300 dimensions respectively by the Stanford team developing GloVe. I decide to use the first model based on the comparison.

Word embedding models generate a vector for each word or each phrase. To have a vector representation for a sentence which composes of a couple of words, one commonly used approach is adding up all the vectors of the words and divide the sum vector by

the count of words as a scaler. The similarity between two vectors can be calculated using cosine similarity. Another approach is to apply Word Movers' Distance [46], which computes the distance (dissimilarity) between two sets of vectors directly. After testing both approaches, I find the Word Movers' Distance is a better choice with much less computational demands. The Word Movers' Distance is computed through an optimization function as shown in Eq. 3.8.

$$
\begin{aligned}
wmd(s_1, s_2) &= \min_{T>=0} \sum_{i=1}^{m} \sum_{j=1}^{n} T_{ij} c(i,j) \\
\text{subject to:} \quad & \sum_{j=1}^{n} T_{ij} = d_i \quad \forall i \in 1, ..., m \\
& \sum_{i=1}^{m} T_{ij} = d_i{}' \quad \forall j \in 1, ..., n
\end{aligned}
\tag{3.8}
$$

where $c(i, j)$ is the travel cost (distance) from word i to word j, which can be an Euclidean distance between their vectors. T_{ij} is a flow matrix denoting how many i in d flow into j in d'. The first constraint guarantees the total outflow from i equals d_i. The second constraint, similarly, ensures the entire inflow to j sums to $d_i{}'$. As a larger distance score implies lower similarity, to have the similarity score of two sentences, I simply take the negative of the distance score, as shown in Eq. 3.9.

$$
Sim(s_1, s_2) = -wmd(s_1, s_2) \tag{3.9}
$$

A Wikipedia page of a candidate location carries richer and more diverse information than what a much shorter test sentence has. Thus, the similarity between the Wikipedia content and the test sentence tends to be diluted if I directly compare the whole Wikipedia content to the test sentence. Therefore, I split the paragraphs into sentences before

computing the similarity score between every sentence of each candidate's Wikipedia page content to the test sentence. The highest similarity score is taken as the matching score of a candidate location to the surface form in a test sentence. The formula is shown in Eq. 3.10.

$$S_{WE}(p \rightarrow c_i) = \max_{\forall s \in S_i} Sim(s, t) \tag{3.10}$$

where S_i refers to a set of sentences in the Wikipedia content for a candidate location c_i. $Sim(s, t)$ computes the similarity score between a sentence s and the sentence t where the surface form p is. A higher matching score is considered as a higher probability that the candidate is the actual place the surface form refers to. This matching score is the final output of the word embedding model.

3.4 Things & Strings Model (TSM)

The entity co-occurrence model makes use of the co-occurrence of entities as contextual clue to disambiguate place names, while the topic model puts emphasis on thematic aspects, namely co-occurring topics. Moreover, the Word embedding model comprehensively capture semantics embedded in the texts, filling the semantic gap between entities and topics in the context. As argued in the introduction, applying a single model, which extracts partial contextual clues, is often not sufficient to differentiate place names from short texts. Thus, I combine the entity-based co-occurrence model with the string-based topic model and word embedding models to an integrated approach called Things & Strings Model (TSM).

The entity co-occurrence model, the topic model, and the word embedding model all return a score when comparing each candidate location with each ambiguous place

name in a sample text. The scores from these three models are, however, not directly comparable. To combine the models, since the scores involve relative probabilistic measures, I must first standardize those scores. This results in setting the standardized mean to zero. Scores originally higher than the mean will be positive, while scores originally lower than the mean will be negative. For each candidate, the standardized scores from the entity co-occurrence model are then combined with the standardized scores from the topic-based model and the standardized scores from the word embedding model along with three weighting parameters, as shown in Eq. 3.11 and 3.12.

$$S_{ETM}(p \to c_i) = \alpha S_{ECM}(p \to c_i) + \beta S_{TM}(p \to c_i) + \gamma S_{WE}(p \to c_i) \tag{3.11}$$

$$\alpha + \beta + \gamma = 1 \tag{3.12}$$

where weighting parameters α, β, and γ are all within the range of [0, 1], and they determine how much each model contributes in the combined approach. I enumerate every combination of α, β, and γ with an interval of 0.01 to find the optimal combination of the weights for the integrated model (details are discussed in Results section). S_{ECM} is the standardized score computed from the entity co-occurrence model for the candidate location c_i with respect to the surface form p. S_{TM} is the standardized score from the topic model, namely the JS divergence. S_{WE} is the standardized score from the the word embedding model. The sum of the weighted standardized scores of these three models is S_{ETM}, the score of the TSM model. Provided that S_{ETM} score has been standardized and it gives a sense of relative likelihood that a candidate to be what the surface form actually refers to, the percentile, instead of an arbitrary number, is used as the threshold

over which candidates are returned as the disambiguation result.

Chapter 4

Evaluation

In this section, I evaluate the performance of the entity co-occurrence model, the topic model, the word embedding model, and the combined TSM. I first describe the methods through which I gathered the testing corpus, and the metrics employed for evaluation, followed by those well recognized named entity disambiguation systems as baselines for comparison. The evaluation results are then displayed and analyzed.

4.1 Preparing the Test Corpus

I constructed a text corpus specifically for the evaluation of my place name disambiguation models, since there is no such corpus available. The corpus is used to evaluate the performance of the proposed models and to compare them to existing systems acting as baselines.

To construct the text corpus, I first derive ambiguous place names from a list of the most common U.S. place names on Wikipedia.[1] As the list also presents the full place names (the state and county names) which could be used to uniquely identify the place of interest, I feed the full place names into the Bing Search API,[2] which returns a set

[1] https://en.wikipedia.org/wiki/List_of_the_most_common_ U.S._place_names
[2] https://datamarket.azure.com/dataset/bing/search

Oxford, Wisconsin — Located in Marquette County in south-central ~~Wisconsin~~, just minutes west of Interstate 39, Oxford invites you to experience our small town charm along with the area's many year-round outdoor attractions.
Jackson, Montana — The tiny town of Jackson, ~~Montana~~ has made a name for itself as a winter sports destination for the adventurous.
Dayton, Nevada — Since the Native-American tribes in the area were nomadic, this made Dayton the first and oldest permanent non-native settlement in ~~Nevada~~.

Table 4.1: Three example records of the test corpus extracted from websites.

of websites related to the place along with URLs. URLs containing "Wikipedia" are filtered out, since my models are trained on the Wikipedia text. I visit those websites and extract those sentences which contain the full place name. The auxiliary part of the full place name (state or county name) is then removed, so the remaining place name is ambiguous. The result of this approach is a set of real-world, i.e., not synthetic, sentences containing ambiguous place names. These sentences comprise the ground truth data.

Sample ground truth sentences are shown in Table 4.1. The full place name and test sentence are separated by an em-dash, and the auxiliary part of the full place name is removed (shown as *striken* for example purposes). It should be noted that the resulting corpus for testing contains noise. Some sentences, for instance, contain no meaningful entities or terms that can be categorized into topics, while others seem to be automatically generated from templates. This noise, however, can help evaluate the robustness of my models. In total, the testing corpus consists of 5,340 sentences, covering 662 unique place entities of 32 ambiguous place names. The average length of a test sentence is 22.5 words with a median of 19. Note that stop words count towards these statistics, while auxiliary parts of the place name do not.

4.2 Metrics

Precision, *recall*, *F-score*, *precision at 1*, and *mean reciprocal rank* are used as the metrics for the performance evaluation of the place name disambiguation models.

The *F-score*, or sometimes called F_1 *score*, is defined as the harmonic mean of precision and recall [47]. In the task of place name disambiguation, precision is a fraction of the count of correctly identified place names and the count of predicting candidate place names in the disambiguation result, and recall is a fraction of the count of correctly identified place names and count of surface forms. Precision and recall are always a trade-off for the task of information retrieval [48]. Both metrics are taken into account when the *F-score* is computed. The *F-score* (See Eq. 4.1) reaches its best value at 1 and worst at 0.

$$F_1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{4.1}$$

For many applications of geographic information retrieval, only one predicting place, if available, is used. Therefore, I also use the metric *precision at 1* (also written as *P@1*), in which only the first predictive location (candidate location with the highest score) is considered.

Although *precision*, *recall*, and *F-score* are widely used for evaluating the performance of information retrieval systems, they fail to capture the order of the results. The *mean reciprocal rank* (MRR), by comparison, takes the order of the results into account. The reciprocal rank of a test sentence is the inverse of the rank of the correctly identified place name $rank_i$ for a surface form. It can be formalized as in Eq. 4.2, where Q is the

set of the predicting candidate place names in the disambiguation result.

$$MRR = \frac{1}{|Q|} \sum_{r=1}^{|Q|} \frac{1}{rank_i} \tag{4.2}$$

4.3 Benchmark systems

DBpedia Spotlight[3], TextRazor[4], and Open Calais[5] were selected as benchmark systems to be compared to the proposed Things & Strings Model.

DBpedia Spotlight is based on DBpedia's rich knowledge base of structured data [24], which is also employed by the proposed model. I implemented a local instance of DBpedia Spotlight Web Service of version 0.7. *Annotate* and *Candidates* are two endpoints of this web service relating to place entity recognition and place name disambiguation. *Candidates* endpoint returns a ranked list of candidates for each recognized entity and concept, while *Annotate* simply returns the best candidate according to the context.

TextRazor and Open Calais are two commercial Web services for named entity recognition and named entity disambiguation. Both services offer application programming interfaces (APIs). The TextRazor API returns only one candidate for each entity recognized from the test sentence. Experiments were conducted [51] to compare several named entity disambiguation systems which included DBpedia Spotlight (V. 0.6, confidence=0, support=0) and TextRazor. In the experiments, TextRazor demonstrates the best performance in terms of F-score.

The Open Calais (Calais - Tagging - RESTful API) web service extracts semantic information from unstructured text developed by Thomson Reuters. Given a text, Open Calais tags text strings with entities of predefined types, including companies, people,

[3]https://github.com/dbpedia-spotlight/dbpedia-spotlight
[4]https://www.textrazor.com/
[5]http://www.opencalais.com/

deals, geographical locations, industries, physical assets, organizations, products, and events. Those tagged entities are meanwhile mapped to Thomson Reuters unique IDs for disambiguation. For the task of place name disambiguation, I only look at those extracted entities whose types are *city*, *provinceOrState*, or *place*. It should be noted that Open Calais API only returns one resolution for each recognized place entities. The API, in addition to tagging entities, assigns multiple soical, topic, and industry tags to each text, describing the content of the text as a whole from different perspectives. The *social tags* contain the resolution of recognized place names. Thus, I include those social tags which mention the ambiguous place name in Open Calais' place name disambiguation result.

4.4 Results

In this section, I present the results of evaluation with comparing the proposed model to the benchmark systems. To differentiate the Things & Strings Model proposed in this thesis which integrates three models from the previously proposed model[23] which integrates two models, I call the new integrated model TSM_3 and the former one TSM_2 in this section.

As discussed in the previous paragraph, the three individual models and the integrated TSM_3 all generate scores which give a sense of relative likelihood that a candidate to be what the surface form actually refers to, so percentile is used as a threshold to determine which candidate locations are included in the final disambiguation result. Generally speaking, with a high threshold, few results are selected, which would lead to a high precision but a low recall; while with a low threshold, many results are returned, which would cause a high recall but a low precision. Such a trade-off between precision and recall can hardly be avoided, so a threshold should be picked wisely. Figure 4.1 demonstrates how precision and recall change along percentiles for the three individual models and

the integrated TSM_3 (with the optimal weight combination; details are discussed below). The typical precision and recall trade-off is clearly shown for all models, except for the entity co-occurrence model, whose curve stays in the range of relatively high recall. This is due to the fact that in over 1000 test sentences, no matched entities is found for any candidate place.

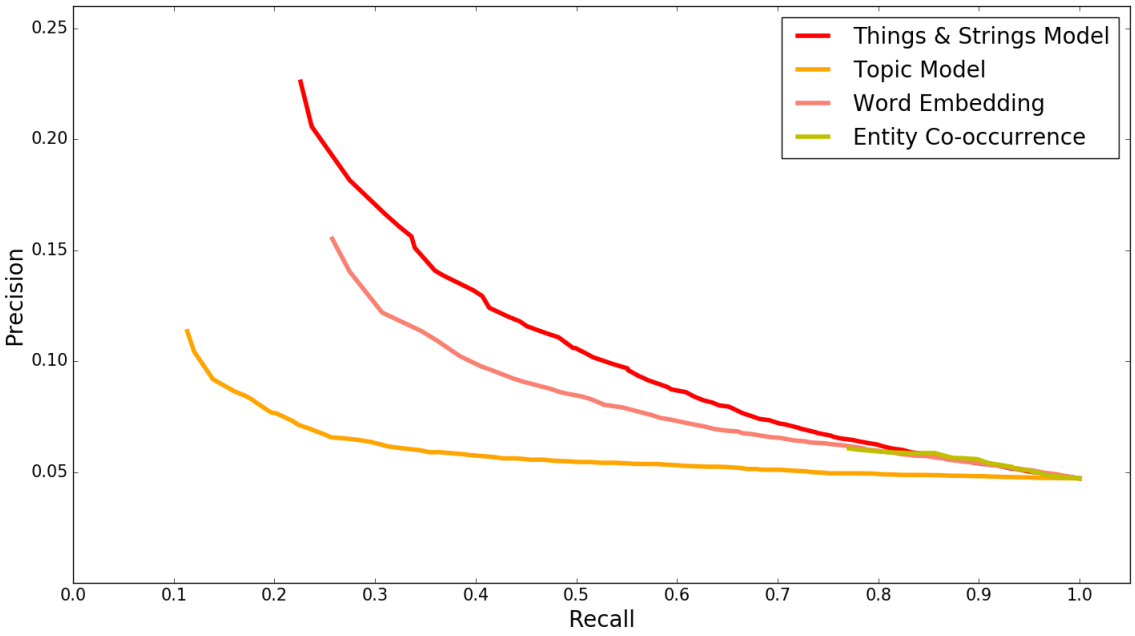

Figure 4.1: Precision and recall curve of entity co-occurrence model, topic model, word embedding model, and TSM_3

Given that TextRazor and Open Calais do not provide controls on how many candidate places are returned and DBpedia Spotlight relies on *Confidence* and *Support* which are not comparable to the percentiles I used as the threshold, I choose the best performance each baseline systems can reach to compare it to my models. To give an example, for DBpedia Spotlight, I picked $Confidence = 0.2$ and $Support = 0$ since this combination of parameters leads to the best overall performance for this setting. From Figure 4.2, I can see that Open Calais can obtain relatively higher F-score than TextRazor and

DBpedia Spotlight on the test corpus. The overall performance of those baseline systems including TSM_2 on the testing dataset is summarized in Table 4.2.

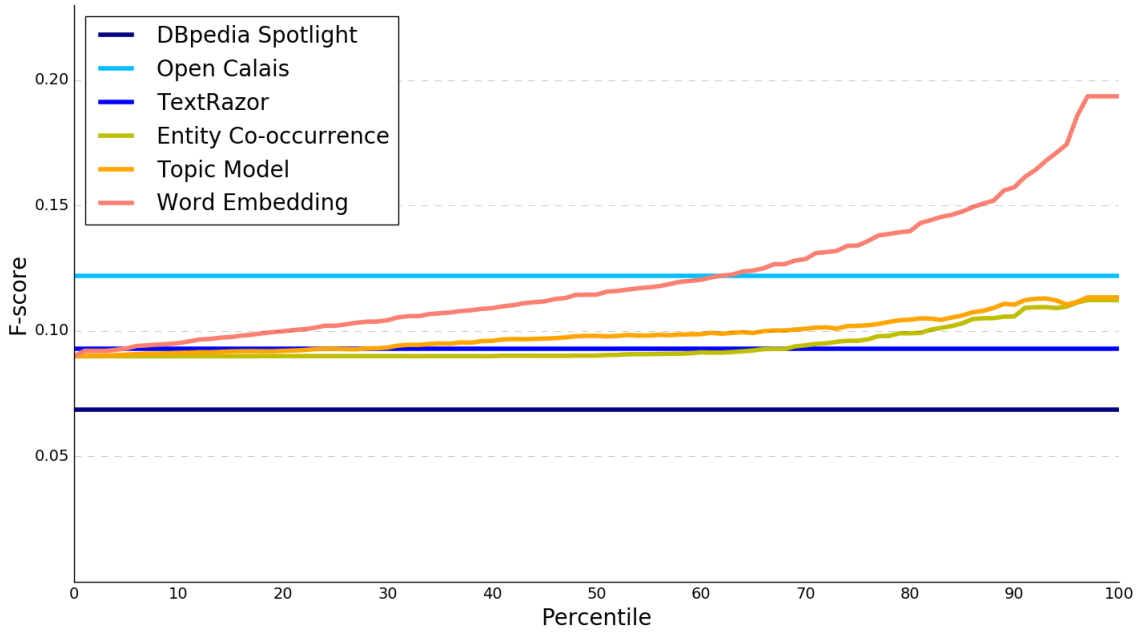

Figure 4.2: F-score for DBpedia Spotlight, TextRazor, Open Calais, entity co-occurrence model, topic model, and word embedding model

Figure 4.2 also shows how F-score of the individual models changes along percentiles. Note that the 90 at the x-axis refers to the 90th percentile, which means that the candidate places with top 10 percentage of scores are selected as the disambiguation result. As shown in the plot, when percentile increases, the F-score of the individual models increases slightly until the 70th percentile when it starts increasing faster. Compared to these baseline systems, the *individual* performance of the entity co-occurrence model and topic model does not show a significant improvement, while the F-score of word embedding model achieves an enhancement, reaching 0.194.

TSM_3, which combines the entity-based co-occurrence model with the topic-based model and word embedding model, employs weighting parameters α, β, and γ to de-

Systems	Parameters	Prec	Recall	F-score
DBpedia Spotlight	$Confidence = 0.2$; $Support = 0$	0.065	0.072	0.069
TextRazor	N/A	0.067	0.15	0.093
Open Calais	N/A	0.105	0.144	0.122
Entity Co-occurrence	97th pctl	0.031	0.771	0.112
Topic Model	97th pctl	0.113	0.113	0.113
Word Embedding	97th pctl	0.155	0.257	0.194
TSM_2	α=0.52 β=0.48 γ=0; 93th pctl	0.132	0.268	0.177
TSM_3	α=0.31 β=0.22 γ=0.47; 97th pctl	0.226	0.226	0.226

Table 4.2: Performance comparison of systems at best performance in terms of Precision, Recall, and F-Score

termine how much each individual model contributes to disambiguation. I test every combination of α, β, and γ, each of which values from 0 to 1 with an interval of 0.01 and meanwhile adheres to the constraint that the weights should always sum to 1. Figure 4.3 shows the highest obtainable F-score with each combination of the weights. I find that when $\alpha = 0.31$, $\beta = 0.22$, and $\gamma = 0.47$, the integrated model yields the highest F-score on the test corpus. This indicates that all three models play essential roles in TSM_3 for disambiguating place name; the word-embedding model contributes fairly more than the other two models. At the 97th percentile, the F-score reaches at 0.226. Out of 5,430 test sentences, ambiguous place names in 1,207 sentences are correctly identified, given the disambiguation result of 5,343 places. It should be noted that F-score, precision and recall of the topic model and TSM_3 are different values though they happen to be rounded to the same values. Figure 4.4 demonstrates how the F-score of TSM_3 ($\alpha = 0.31$, $\beta = 0.22$, and $\gamma = 0.47$) changes along percentile and its comparison to the baseline systems. Although the word embedding model individually outperforms the benchmarks, with the support of entity co-occurrence model and topic model, the integrated TSM_3 is able to achieve even higher F-score, which almost double the topmost F-score the benchmark systems can obtain.

In many cases, only one candidate location (if available) is picked as the disambigua-

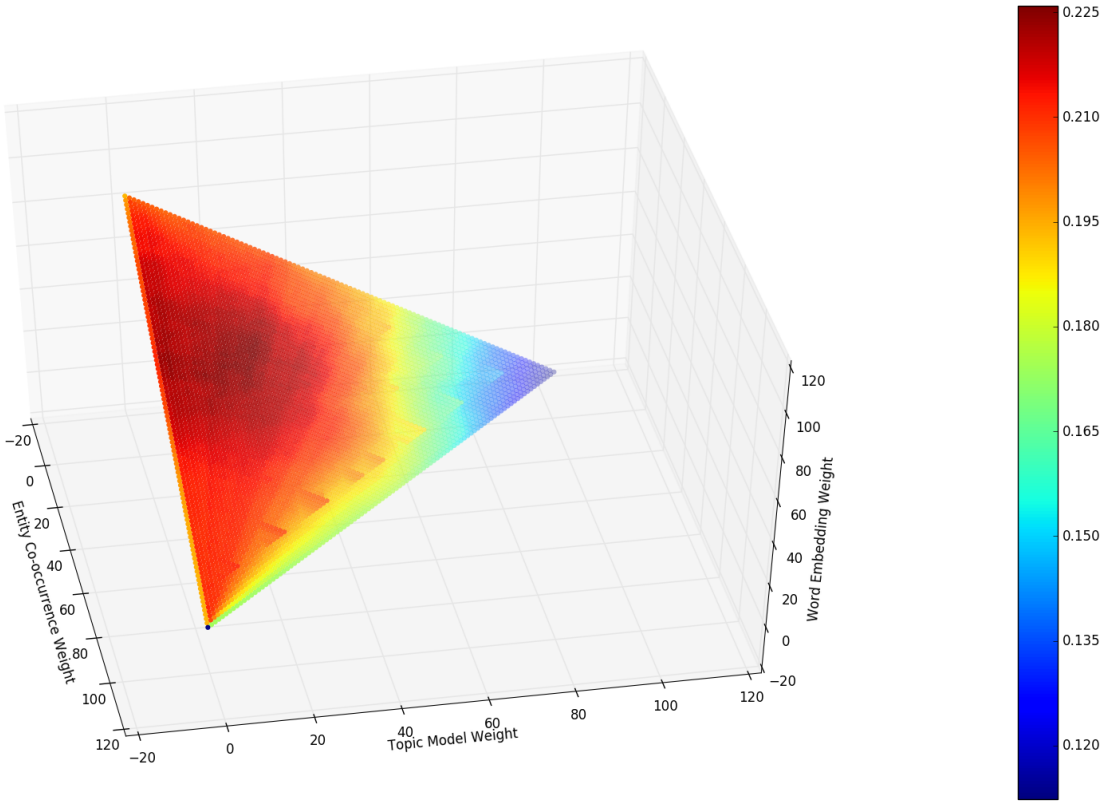

Figure 4.3: F-score of TSM_3 across the combinations of α, β, and γ

tion result, so I also compute Precision at 1 to evaluate the models. Figure 4.5 shows P@1 of TSM_3 with every combination of the weights. The peak value of P@1 is 0.226, found at the point where $\alpha = 0.31$, $\beta = 0.22$, and $\gamma = 0.47$, is exactly the same peak point as in Figure 4.3. Comparing to Figure 4.3, it demonstrates similar pattern with higher values concentrating around the peak point, and when entity co-occurrence model holds more weight, the precision tends to decrease.

As stated in the previous paragraph, TextRazor only outputs at most one result, so does the DBpedia Spotlight Web Service in the *Annotation* mode. For Open Calais, the disambiguation result is ranked, so the first returned result is taken for this evaluation.

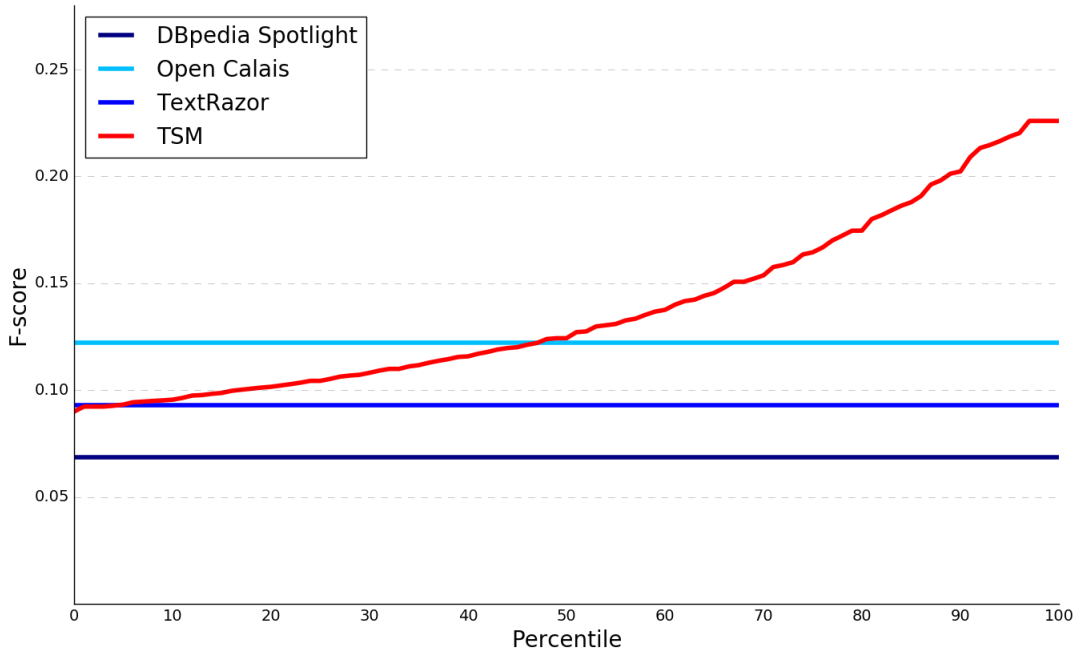

Figure 4.4: F-score of DBpedia Spotlight, TextRazor, Open Calais, and Things & Strings Model (TSM_3)

Table 4.3 lists P@1, true positives, and estimated positives of TSM_3, TSM_2, the individual models and the benchmark systems when only one candidate location is taken in each test sentence's disambiguation result. True positives (TF) refers to the count of correctly identified place names, and estimated positives ($\hat{p}$) is the count of the predicting candidate locations in disambiguation result. The division of true positives by estimated positives is the precision.

For Mean Reciprocal Rank (MRR), TSM_3, with α=0.31, β=0.22, γ=0.47, tops 0.373, beating DBpedia Spotlight and TextRazor, and TSM_2 which get 0.054, 0.076, and 0.310 respectively.

Overall, based on the evaluation, the proposed TSM_3 substantially outperforms existing named entity disambiguation systems in terms of F-score, P@1, and MRR. The word embedding model alone performs better than all the benchmark systems, while a

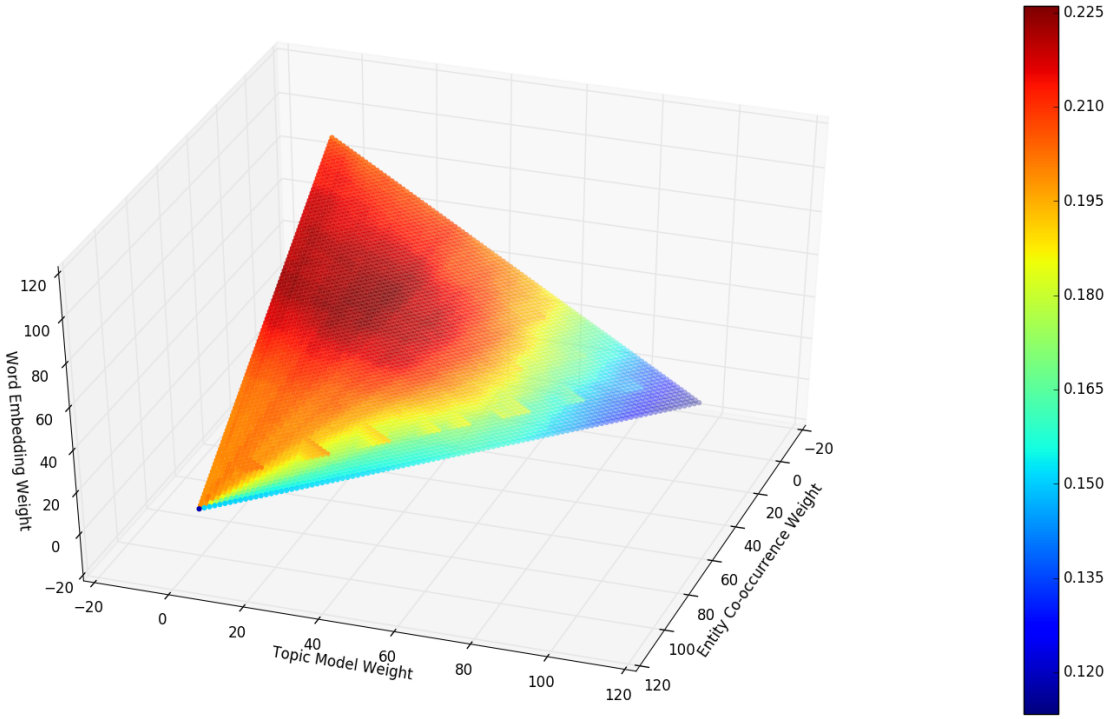

Figure 4.5: P@1 of TSM_3 across the combinations of α, β, and γ

combination of the entity co-occurrence model, the topic model, and the word embedding model, namely the TSM_3, can achieve even better performance. In terms of F-score, TSM_3 outperforms the benchmark systems by up to 85%, and in terms of P@1, the integrated model exceeds them by up to 46%. The fact that all F-scores are relatively low is an important reminder for the facts that place name disambiguation from short texts is a difficult task and that some test sentences did not contain no or only minimal contextual clues.

Systems	Parameters	TP	$\hat{P}$	P@1
DBpedia Spotlight	$Conf = 0.2$; $Sup = 0$	232	3741	0.062
TextRazor	N/A	147	4755	0.031
Open Calais	N/A	463	2959	0.156
Entity Co-occurrence	N/A	630	5340	0.118
Topic Model	N/A	605	5340	0.113
Word Embedding	N/A	1087	5340	0.204
TSM_2	α=0.47, β=0.53, γ=0	848	5340	0.159
TSM_3	α=0.31, β=0.22, γ=0.47	1207	5340	0.227

Table 4.3: Performance comparison of systems when only one candidate location is taken in each test sentence's disambiguation result in terms of True Positives (TP), Estimated Positives ($\hat{p}$), and P@1

Chapter 5

Discussion and Conclusions

To conclude, I propose a novel approach to tackle the challenging task of disambiguating place names from short texts. Place name disambiguation is an important part of knowledge extraction and a core component of geographic information retrieval systems. I have presented three individual models that are driven by different perspectives, namely an entity-based co-occurrence model, a topic-based model, and a word embedding model. The first model focuses on the semantic connections between entities and thereby on *things*, while the other two models work on the linguistic level by investigating topics and semantics embedded in the text associated with places and thereby *strings*. The integration of these three models, the Things & Strings Model (TSM), demonstrates a substantially better performance than the used benchmark systems with respect to F-score, precision at 1, and MRR.

The place name, or toponym, is an essential component of a text, specifying the location where the things or events discussed in the text take place. It also helps with identifying other entities in the text, so it is key for machines to "understand" the text.

Wikipedia and DBpedia are used as the training data for the models I build, providing the ground truth descriptions for places. Formating its rich knowledge as linked data, DBpedia allows us to readily make use of a place entity's properties and its related entities. However, DBpedia lacks descriptive sentences where richer semantics are hidden,

which are also useful for place name disambiguation. Wikipedia filled the gap, rendering more abundant and complex knowledge in the human language. However, the natural language representation of Wikipedia articles is a mixture of various perspectives of a place. The noise information irrelevant to a place could also be found in Wikipedia. My approach takes advantage of the merits of both sources and alleviate the influence from their shortcomings.

The improvement of place name disambiguation could facilitate more research, especially in the field of Geographic Information Science. Being able to accurately identifying place names, geographic information buried in the text could be uncovered. To give a simple example, in the past, I only knew the location of a Twitter tweet if it was attached with coordinates, but now I can map the places the tweets are talking about. It would be interesting to see how people in one city mention other cities in the tweets. Moreover, It could also expedite spatial data sharing in the community, which is key to the researchers and users of GIS. A good spatial data sharing platform would rely on the performance of its data searching/query function [52, 53]. The current query function is merely about word or phrase matching, but if a user is looking for census block data of Isla Vista, most likely few results would be returned. However, with name recognition of Isla Vista and knowing Isla Vista is a subregion of Santa Barbara county based on a knowledge graph, like DBpedia, the user is able to find census block data for Santa Barbara county which includes the target Isla Vista census block data. Additionally, this research could bridge place-based GIS and space-based GIS.

A more accurate place name resolver can benefit many applications, improving user experience. The proposed method could be applied to many location-based service applications. In a navigation application, an accurate place name resolver allows users to input a common name of a place, instead of the full name of the place. The users' metadata, such as the users' current location, history of previous queries, and recent posts on

social media, could be utilized as the context to help disambiguate the place name.

Nonetheless, there is space for future improvements. For the entity co-occurrence model, properties other than those with the namespaces of *dbo* and *dbp* have been filtered out. The same is true for literals. Both of them could be added to a future version of TSM, although they would require more work on the used similarity functions in case of the literals and a better alignment to ensure that properties from different namespaces are not mere duplicates. I have used LDA for topic modeling but this is not the only choice that can be used to learn a topic model so other approaches will be tested in the future. As for the word embedding, doc2vec [54] could be tried out to see if it can performance better than the combination of word2vec and Word's Mover Distance. Moreover, TSM is realized as a relatively simple convex combination of three individual models. Other approaches could be investigated as well.

Similar to many other machine learning based models, there is no formal explanation of why the word2vec model leads to good word representations and has better performance for many tasks [55]. For this research, I think word2vec does well in capturing the semantic and relationship between words in the texts more comprehensively than the other models. However, which semantic and which relationship are captured is unknown, and what is missed by word embedding model but gets caught by the entity co-occurrence model and topic model is also not clear.

The performance of my proposed method relies on the comprehensiveness of candidate locations and the availability of DBpedia descirption of the candidate locations. In other word, if a place is not covered by DBpedia or there is no Wikipedia article about it, the model will wrongly pick another candidate location. This could be solved by a more well-designed mechanism to set thresholds, so if all scores of candidate locations are below the threshold, "not found" would be returned, instead of accepting an answer close to the threshold. Additionally, in digital humanity, many ancient place name should

be identified, the proposed model cannot perform without a list of candidate ancient locations and a description of them.

As for the experiment, although place entities in my testing corpus have highly ambiguous place names, those places are all some kind of administrative divisions (i.e., cities, towns, villages, etc.) and located within the United States. A potential improvement could be seeking more ambiguous place names from other types of places which are outside of the United States.

Bibliography

[1] R. Purves and C. Jones, *Geographic information retrieval*, *SIGSPATIAL Special* **3** (2011), no. 2 2–4.

[2] K. Janowicz and P. Hitzler, *The digital earth as knowledge engine*, *Semantic Web* **3** (2012), no. 3 213–221.

[3] B. Adams, G. McKenzie, and M. Gahegan, *Frankenplace: interactive thematic mapping for ad hoc exploratory search*, in *Proceedings of the 24th International Conference on World Wide Web*, pp. 12–22, ACM, 2015.

[4] M. F. Goodchild and J. A. Glennon, *Crowdsourcing geographic information for disaster response: a research frontier*, *International Journal of Digital Earth* **3** (2010), no. 3 231–241.

[5] C. B. Jones and R. S. Purves, *Geographical information retrieval*, *International Journal of Geographical Information Science* **22** (2008), no. 3 219–228.

[6] R. C. Bunescu and M. Pasca, *Using encyclopedic knowledge for named entity disambiguation.*, in *EACL*, vol. 6, pp. 9–16, 2006.

[7] A. Fader, S. Soderland, O. Etzioni, and T. Center, *Scaling wikipedia-based named entity disambiguation to arbitrary web text*, in *Proceedings of the IJCAI Workshop on User-contributed Knowledge and Artificial Intelligence: An Evolving Synergy, Pasadena, CA, USA*, pp. 21–26, 2009.

[8] A. Spitz, J. Geiß, and M. Gertz, *So far away and yet so close: Augmenting toponym disambiguation and similarity with text-based networks*, in *Proceedings of the Third International ACM SIGMOD Workshop on Managing and Mining Enriched Geo-Spatial Data*, GeoRich '16, (New York, NY, USA), pp. 2:1–2:6, ACM, 2016.

[9] Y. Hu, K. Janowicz, and S. Prasad, *Improving wikipedia-based place name disambiguation in short texts using structured data from dbpedia*, in *Proceedings of the 8th Workshop on Geographic Information Retrieval*, p. 8, ACM, 2014.

[10] S. Overell and S. Rüger, *Using co-occurrence models for placename disambiguation*, *International Journal of Geographical Information Science* **22** (2008), no. 3 265–287.

[11] B. Adams and K. Janowicz, *On the geo-indicativeness of non-georeferenced text.*, in *International AAAI Conference on Web and Social Media (ICWSM)*, pp. 375–378, 2012.

[12] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, *Distributed representations of words and phrases and their compositionality*, in *Advances in neural information processing systems*, pp. 3111–3119, 2013.

[13] S. Cucerzan, *Large-scale named entity disambiguation based on wikipedia data.*, in *EMNLP-CoNLL*, vol. 7, pp. 708–716, 2007.

[14] D. Milne and I. H. Witten, *Learning to link with wikipedia*, in *Proceedings of the 17th ACM conference on Information and knowledge management*, pp. 509–518, ACM, 2008.

[15] R. Mihalcea and A. Csomai, *Wikify!: linking documents to encyclopedic knowledge*, in *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pp. 233–242, ACM, 2007.

[16] Z. Zheng, F. Li, M. Huang, and X. Zhu, *Learning to link entities with knowledge base*, in *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 483–491, Association for Computational Linguistics, 2010.

[17] J. L. Leidner, *Toponym resolution in text: annotation, evaluation and applications of spatial grounding*, in *ACM SIGIR Forum*, vol. 41, pp. 124–126, ACM, 2007.

[18] I. M. R. Machado, R. O. de Alencar, R. de Oliveira Campos Jr, and C. A. Davis Jr, *An ontological gazetteer and its application for place name disambiguation in text*, *Journal of the Brazilian Computer Society* **17** (2011), no. 4 267–279.

[19] M. D. Lieberman and H. Samet, *Adaptive context features for toponym resolution in streaming news*, in *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pp. 731–740, ACM, 2012.

[20] M. Speriosu and J. Baldridge, *Text-driven toponym resolution using indirect supervision.*, in *ACL (1)*, pp. 1466–1476, 2013.

[21] W. Zhang and J. Gelernter, *Geocoding location expressions in twitter messages: A preference learning method*, *Journal of Spatial Information Science* **2014** (2014), no. 9 37–70.

[22] J. de Bruijn, H. de Moel, B. Jongman, J. Wagemaker, and J. C. J. H. Aerts, *Taggs: Grouping tweets to improve global geotagging for disaster response*, *Natural Hazards and Earth System Sciences Discussions* **2017** (2017) 1–22.

[23] Y. Ju, B. Adams, K. Janowicz, Y. Hu, B. Yan, and G. McKenzie, *Things and strings: Improving place name disambiguation from short texts by combining entity co-occurrence with topic modeling*, in *Knowledge Engineering and Knowledge Management: 20th International Conference, EKAW 2016, Bologna, Italy, November 19-23, 2016, Proceedings 20*, pp. 353–367, Springer, 2016.

[24] P. N. Mendes, M. Jakob, A. García-Silva, and C. Bizer, *Dbpedia spotlight: shedding light on the web of documents*, in *Proceedings of the 7th international conference on semantic systems*, pp. 1–8, ACM, 2011.

[25] X. Han and J. Zhao, *Structural semantic relatedness: a knowledge-based method to named entity disambiguation*, in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pp. 50–59, Association for Computational Linguistics, 2010.

[26] D. M. Blei, A. Y. Ng, and M. I. Jordan, *Latent Dirichlet allocation*, *Journal of Machine Learning Research* **3** (2003), no. Jan 993–1022.

[27] K. Sahr, D. White, and A. J. Kimerling, *Geodesic discrete global grid systems*, *Cartography and Geographic Information Science* **30** (2003), no. 2 121–134.

[28] R. W. Gray, *Exact transformation equations for Fuller's world map*, *Cartographica: The International Journal for Geographic Information and Geovisualization* **32** (1995), no. 3 17–25.

[29] A. K. McCallum, *Mallet: A machine learning for language toolkit*, .

[30] H. M. Wallach, D. M. Mimno, and A. McCallum, *Rethinking lda: Why priors matter*, in *Advances in neural information processing systems*, pp. 1973–1981, 2009.

[31] M. Steyvers and T. Griffiths, *Probabilistic topic models*, *Handbook of latent semantic analysis* **427** (2007), no. 7 424–440.

[32] Z. S. Harris, *Distributional structure*, *Word* **10** (1954), no. 2-3 146–162.

[33] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, *Learning internal representations by error propagation*, tech. rep., California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[34] G. E. Hinton, *Learning distributed representations of concepts*, in *Proceedings of the eighth annual conference of the cognitive science society*, vol. 1, p. 12, Amherst, MA, 1986.

[35] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, *A neural probabilistic language model*, *Journal of machine learning research* **3** (2003), no. Feb 1137–1155.

[36] R. Collobert and J. Weston, *A unified architecture for natural language processing: Deep neural networks with multitask learning*, in *Proceedings of the 25th international conference on Machine learning*, pp. 160–167, ACM, 2008.

[37] J. Turian, L. Ratinov, and Y. Bengio, *Word representations: a simple and general method for semi-supervised learning*, in *Proceedings of the 48th annual meeting of the association for computational linguistics*, pp. 384–394, Association for Computational Linguistics, 2010.

[38] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, *Natural language processing (almost) from scratch*, *Journal of Machine Learning Research* **12** (2011), no. Aug 2493–2537.

[39] H. Schwenk, *Continuous space language models*, *Computer Speech & Language* **21** (2007), no. 3 492–518.

[40] J. Weston, S. Bengio, and N. Usunier, *Wsabie: Scaling up to large vocabulary image annotation*, in *IJCAI*, vol. 11, pp. 2764–2770, 2011.

[41] X. Glorot, A. Bordes, and Y. Bengio, *Domain adaptation for large-scale sentiment classification: A deep learning approach*, in *Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 513–520, 2011.

[42] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, *Indexing by latent semantic analysis*, *Journal of the American society for information science* **41** (1990), no. 6 391.

[43] T. Mikolov, K. Chen, G. Corrado, and J. Dean, *Efficient estimation of word representations in vector space*, *arXiv preprint arXiv:1301.3781* (2013).

[44] J. Pennington, R. Socher, and C. D. Manning, *Glove: Global vectors for word representation.*, in *EMNLP*, vol. 14, pp. 1532–1543, 2014.

[45] R. Řehůřek and P. Sojka, *Software Framework for Topic Modelling with Large Corpora*, in *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, (Valletta, Malta), pp. 45–50, ELRA, May, 2010. `http://is.muni.cz/publication/884893/en`.

[46] M. Kusner, Y. Sun, N. Kolkin, and K. Weinberger, *From word embeddings to document distances*, in *International Conference on Machine Learning*, pp. 957–966, 2015.

[47] M. Bilenko, R. Mooney, W. Cohen, P. Ravikumar, and S. Fienberg, *Adaptive name matching in information integration*, *IEEE Intelligent Systems* **18** (2003), no. 5 16–23.

[48] M. Buckland and F. Gey, *The relationship between recall and precision*, *Journal of the American society for information science* **45** (1994), no. 1 12.

[49] M. Karimzadeh, *Performance evaluation measures for toponym resolution*, in *Proceedings of the 10th Workshop on Geographic Information Retrieval*, p. 8, ACM, 2016.

[50] P. Resnik and D. Yarowsky, *Distinguishing systems and distinguishing senses: New evaluation methods for word sense disambiguation*, *Natural language engineering* **5** (1999), no. 2 113–133.

[51] G. Rizzo, M. van Erp, and R. Troncy, *Benchmarking the extraction and disambiguation of named entities on the semantic web.*, in *LREC*, pp. 4593–4600, 2014.

[52] Y. Hu, K. Janowicz, S. Prasad, and S. Gao, *Enabling semantic search and knowledge discovery for arcgis online: a linked-data-driven approach*, in *Geographic Information Science as an Enabler of Smarter Cities*, pp. 107–124, Springer, 2015.

[53] S. Lafia, J. Jablonski, W. Kuhn, S. Cooley, and F. A. Medrano, *Spatial discovery and the research library*, *Transactions in GIS* **20** (2016), no. 3 399–412.

[54] Q. Le and T. Mikolov, *Distributed representations of sentences and documents*, in *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pp. 1188–1196, 2014.

[55] Y. Goldberg and O. Levy, *word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method*, *arXiv preprint arXiv:1402.3722* (2014).